SQL PROJECT (ADHOC Analysis)

Perfect, the PDF is now accessible 👍
Below is a **clear, step-by-step solution with SQL queries** for the **Codebasics – Ad Hoc SQL Requests** using the **Atliq Hardware schema**, exactly as asked in the PDF .

I'll follow this structure for **each question**:

1. Business understanding
2. Tables involved
3. Logic (step-by-step)
4. Final SQL query

Q1. Markets where "Atliq Exclusive" operates in APAC

1️⃣ Business Requirement

Find all **markets** in the **APAC region** where the customer **Atliq Exclusive** does business.

2️⃣ Tables Used

- `dim_customer`

3️⃣ Logic

- Filter customer = `'Atliq Exclusive'`
- Filter region = `'APAC'`
- Select distinct markets

4️⃣ SQL Query

```
SELECT DISTINCT market
FROM dim_customer
```

```
WHERE customer = 'Atliq Exclusive'
  AND region = 'APAC';
```

Q2. % Increase of Unique Products in 2021 vs 2020

1️⃣ Business Requirement

Compare **unique products sold** in 2020 and 2021 and calculate the percentage increase.

2️⃣ Tables Used

- `fact_sales_monthly`

3️⃣ Logic

- Count distinct `product_code` for 2020
- Count distinct `product_code` for 2021
- Apply percentage formula

4️⃣ SQL Query

```
SELECT
    COUNT(DISTINCT CASE WHEN fiscal_year = 2020 THEN
product_code END) AS unique_products_2020,
    COUNT(DISTINCT CASE WHEN fiscal_year = 2021 THEN
product_code END) AS unique_products_2021,
    ROUND(
        (
            COUNT(DISTINCT CASE WHEN fiscal_year = 2021 THEN
product_code END)
          - COUNT(DISTINCT CASE WHEN fiscal_year = 2020 THEN
product_code END)
        ) * 100.0
```

```
        / COUNT(DISTINCT CASE WHEN fiscal_year = 2020 THEN
product_code END)
    , 2) AS percentage_chg
FROM fact_sales_monthly;
```

Q3. Unique Product Count per Segment

1️⃣ Business Requirement

Show how many **unique products** exist in each **segment**.

2️⃣ Tables Used

- `dim_product`

3️⃣ Logic

- Group by `segment`
- Count distinct `product_code`
- Sort descending

4️⃣ SQL Query

```
SELECT
    segment,
    COUNT(DISTINCT product_code) AS product_count
FROM dim_product
GROUP BY segment
ORDER BY product_count DESC;
```

Q4. Segment with Highest Increase in Products (2021 vs 2020)

1️⃣ Business Requirement

Identify which **segment grew the most** in terms of unique products.

2️⃣ Tables Used

- `fact_sales_monthly`
- `dim_product`

3️⃣ Logic

- Join sales with product
- Count unique products per segment for 2020 & 2021
- Calculate difference

4️⃣ SQL Query

```
SELECT
    p.segment,
    COUNT(DISTINCT CASE WHEN s.fiscal_year = 2020 THEN
s.product_code END) AS product_count_2020,
    COUNT(DISTINCT CASE WHEN s.fiscal_year = 2021 THEN
s.product_code END) AS product_count_2021,
    COUNT(DISTINCT CASE WHEN s.fiscal_year = 2021 THEN
s.product_code END) -
    COUNT(DISTINCT CASE WHEN s.fiscal_year = 2020 THEN
s.product_code END) AS difference
FROM fact_sales_monthly s
JOIN dim_product p
    ON s.product_code = p.product_code
GROUP BY p.segment
ORDER BY difference DESC;
```

Q5. Products with Highest & Lowest Manufacturing Cost

1️⃣ Business Requirement

Find products with **max and min manufacturing cost**.

2️⃣ Tables Used

- `fact_manufacturing_cost`
- `dim_product`

3️⃣ Logic

- Join cost and product tables
- Filter using MAX and MIN cost

4️⃣ SQL Query

```
SELECT
    p.product_code,
    p.product,
    m.manufacturing_cost
FROM fact_manufacturing_cost m
JOIN dim_product p
    ON m.product_code = p.product_code
WHERE m.manufacturing_cost = (SELECT MAX(manufacturing_cost)
FROM fact_manufacturing_cost)
   OR m.manufacturing_cost = (SELECT MIN(manufacturing_cost)
FROM fact_manufacturing_cost);
```

Good catch — you're right 👍
Below is a **clear, step-by-step explanation for Questions 6 to 10**,

written in a **beginner-friendly + interview-ready** manner, aligned with the PDF requirements .

I'll follow the same structured approach:

1. Business question
2. Tables used
3. Step-by-step logic (plain English)
4. Final SQL query (same as before, but now explained)

Q6. Top 5 customers by average discount (India, FY 2021)

1️⃣ Business Question

Management wants to know:

> Which **Indian customers** received the **highest average pre-invoice discount** in **fiscal year 2021**?

This helps identify **highly discounted customers** and review pricing strategy.

2️⃣ Tables Used

Table	Purpose
`fact_pre_invoice_deductions`	Contains discount percentages
`dim_customer`	Customer details & market

3️⃣ Step-by-Step Logic

1. Use `fact_pre_invoice_deductions` to get `pre_invoice_discount_pct`
2. Join with `dim_customer` to get customer names and market
3. Filter:
    - `fiscal_year = 2021`
    - `market = 'India'`
4. Calculate **average discount** per customer
5. Sort by average discount (highest first)
6. Return **top 5 customers**

4️⃣ SQL Query

```
SELECT
    c.customer_code,
    c.customer,
    ROUND(AVG(d.pre_invoice_discount_pct), 2) AS
average_discount_percentage
FROM fact_pre_invoice_deductions d
JOIN dim_customer c
    ON d.customer_code = c.customer_code
WHERE d.fiscal_year = 2021
  AND c.market = 'India'
GROUP BY c.customer_code, c.customer
ORDER BY average_discount_percentage DESC
LIMIT 5;
```

Q7. Monthly Gross Sales for “Atliq Exclusive”

1️⃣ Business Question

The business wants to:

> Analyze **monthly gross sales trends** for **Atliq Exclusive** to identify **high and low performing months**.

2️⃣ Tables Used

Table	Purpose
`fact_sales_monthly`	Sold quantities by date
`fact_gross_price`	Product prices
`dim_customer`	Customer name

3️⃣ Step-by-Step Logic

1. Filter customer = **Atliq Exclusive**
2. Join sales with gross price to calculate sales value

Multiply:
sold_quantity × gross_price

3.
4. Extract **Month** and **Year** from the date
5. Aggregate sales at month level
6. Sort chronologically

4️⃣ SQL Query

```
SELECT
    MONTHNAME(s.date) AS Month,
    YEAR(s.date) AS Year,
    ROUND(SUM(s.sold_quantity * g.gross_price), 2) AS
gross_sales_amount
FROM fact_sales_monthly s
JOIN dim_customer c ON s.customer_code = c.customer_code
JOIN fact_gross_price g ON s.product_code = g.product_code
WHERE c.customer = 'Atliq Exclusive'
GROUP BY Year, Month
ORDER BY Year, MONTH(s.date);
```

Q8. Quarter in 2020 with Maximum Sold Quantity

1️⃣ Business Question

> In which **quarter of 2020** did Atliq Hardware sell the **highest quantity**?

This helps in **seasonality analysis**.

2️⃣ Tables Used

Table
`fact_sales_monthly`

3️⃣ Step-by-Step Logic

1. Filter data for `fiscal_year = 2020`
2. Convert month to quarter using `QUARTER(date)`
3. Sum `sold_quantity` per quarter
4. Sort quarters by total quantity (descending)

4️⃣ SQL Query

```
SELECT
    CONCAT('Q', QUARTER(date)) AS Quarter,
    SUM(sold_quantity) AS total_sold_quantity
FROM fact_sales_monthly
WHERE fiscal_year = 2020
GROUP BY Quarter
ORDER BY total_sold_quantity DESC;
```

Q9. Channel Contribution to Gross Sales (FY 2021)

1️⃣ Business Question

> Which **sales channel** contributed the **most to gross sales** in FY 2021, and what is its **percentage contribution**?

2️⃣ Tables Used

Table	Purpose
`fact_sales_monthly`	Sales quantity

`fact_gross_price`	Product pricing
`dim_customer`	Channel information

3️⃣ Step-by-Step Logic

1. Filter sales for `fiscal_year = 2021`
2. Calculate gross sales = quantity × price
3. Aggregate sales by **channel**
4. Convert sales to **millions**
5. Calculate **percentage contribution** using window function

---

4️⃣ SQL Query

```
SELECT
    c.channel,
    ROUND(SUM(s.sold_quantity * g.gross_price) / 1000000, 2) AS
gross_sales_mln,
    ROUND(
        SUM(s.sold_quantity * g.gross_price) * 100.0 /
        SUM(SUM(s.sold_quantity * g.gross_price)) OVER (), 2
    ) AS percentage
FROM fact_sales_monthly s
JOIN dim_customer c ON s.customer_code = c.customer_code
JOIN fact_gross_price g ON s.product_code = g.product_code
WHERE s.fiscal_year = 2021
GROUP BY c.channel;
```

Q10. Top 3 Products per Division (FY 2021)

1️⃣ Business Question

> Identify the **top 3 best-selling products** (by quantity) in **each division** for FY 2021.

2️⃣ Tables Used

Table
`fact_sales_monthly`
`dim_product`

3️⃣ Step-by-Step Logic

1. Filter sales for `fiscal_year = 2021`
2. Aggregate total sold quantity per product
3. Partition data by **division**
4. Rank products within each division using `RANK()`
5. Select only **top 3 ranks**

4️⃣ SQL Query

```
SELECT *
FROM (
    SELECT
        p.division,
        s.product_code,
```

```
        p.product,
        SUM(s.sold_quantity) AS total_sold_quantity,
        RANK() OVER (PARTITION BY p.division ORDER BY
SUM(s.sold_quantity) DESC) AS rank_order
    FROM fact_sales_monthly s
    JOIN dim_product p ON s.product_code = p.product_code
    WHERE s.fiscal_year = 2021
    GROUP BY p.division, s.product_code, p.product
) ranked
WHERE rank_order <= 3;
```

🔑 **Interview Tip**

If asked **“Why window functions?”** in Q9 or Q10:

> Window functions allow us to calculate totals and rankings **without collapsing rows**, making percentage and ranking logic efficient and readable.
